

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра компьютерных систем в управлении и проектировании (КСУП)

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ POSTGRESQL

Отчет по лабораторной работе по дисциплине «Основы разработки баз
данных»

Студент гр. 573-3

_____ Р.В. Слиньков
дата

подпись
Руководитель:

Преподаватель:

_____ Р. О. Остапенко
оценка _____ подпись

дата

Томск, 2025

Введение

В современном мире базы данных играют ключевую роль в системах хранения и обработки информации. Они используются в самых разных областях — от небольших веб-приложений до крупных корпоративных систем. Эффективная организация данных позволяет оптимизировать работу программных продуктов, обеспечить целостность и безопасность информации, а также упростить её поиск и анализ.

Цель данной лабораторной работы — познакомиться с основными принципами проектирования реляционных баз данных, освоить синтаксис языка структурированных запросов SQL и научиться создавать таблицы, определять связи между ними, а также накладывать ограничения для обеспечения корректности данных. В процессе выполнения работы будет спроектирована и реализована структура базы данных на языке SQL в среде PostgreSQL, отражающая реальные сущности и их взаимосвязи.

Задачи:

Реализовать БД согласно требованиям:

1. БД должна быть реляционной;
2. БД должна быть нормализована по 3НФ (или БКНФ);
3. БД должна состоять минимум из 3-4 связанных таблиц (на каждого студента); в таблицах БД должны быть наложены ограничения на поля таблиц (по усмотрению разработчиков решается, на какие поля будут наложены ограничения). Должно быть не менее 3 ограничений на разные поля таблиц (без учета ограничений PRIMARY KEY, FOREIGN KEY). Ограничения проверки, значения по умолчанию обязательно;
4. должна быть обеспечена ссылочная целостность;
5. В БД должны быть обязательно поля разных типов данных, в том числе поля:
 - a. Позволяющие хранить рисунки;
 - b. Позволяющие хранить длинный текст;

- с. Значения, в которых должны соответствовать списку значений, например Пол мужской или женский;
- d. Значения логического типа.

1 ОСНОВНАЯ ЧАСТЬ

1.1 Описание данных для реализации БД

В ходе индивидуальных заданий была разработана реляционная модель данных (рисунок 1.1), которую необходимо реализовать в виде БД.

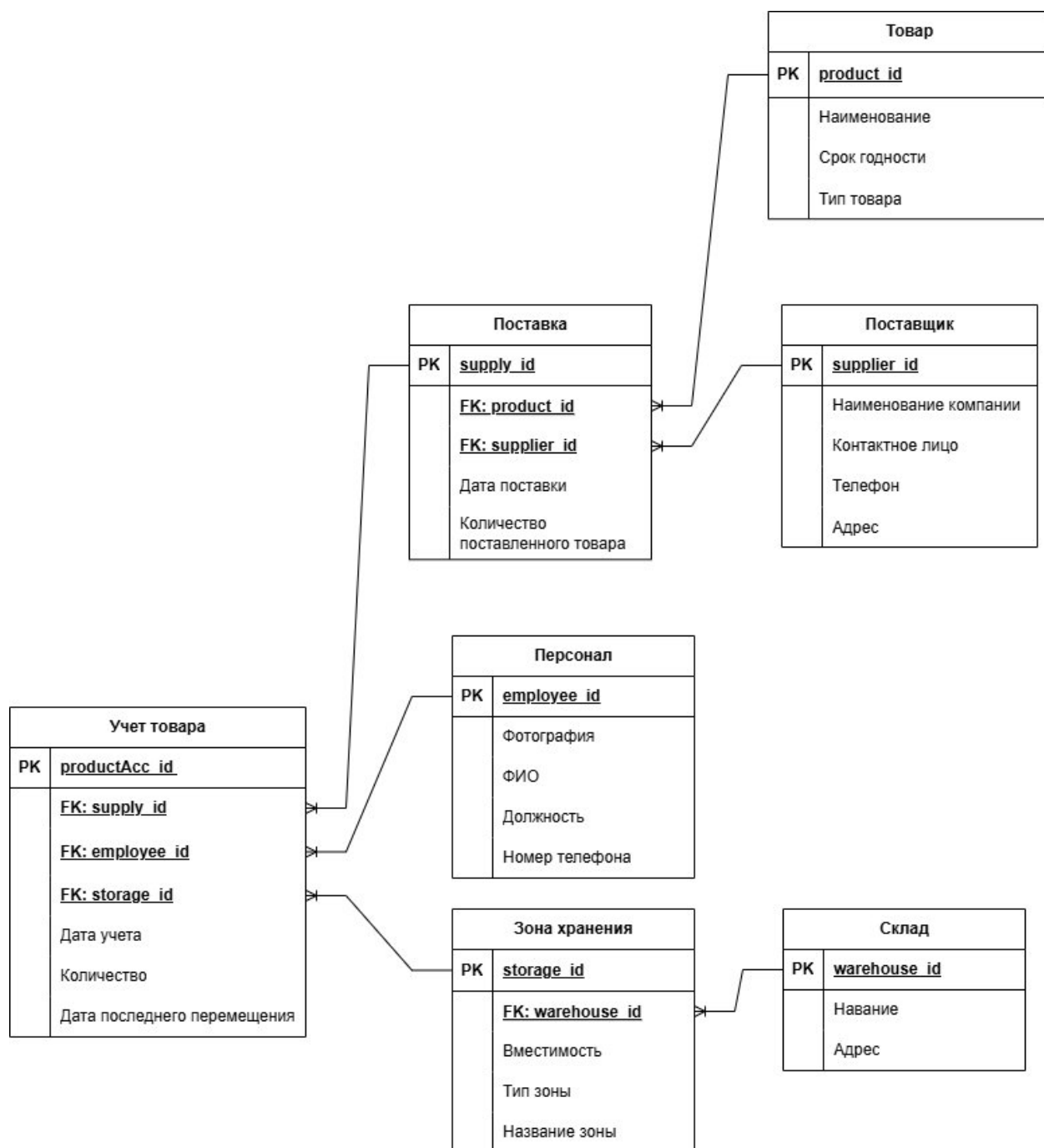


Рисунок 1.1 – Реляционная модель данных

Опишем сущности в виде таблиц, чтобы потом перенести их в БД. Также при необходимости добавим поля и ограничения, чтобы они соответствовали формулировке задания.

Для «Склад» (таблица 1.1) были описаны характеристики.

Таблица 1.1 – Описание «Склад»

Имя поля	Тип данных	Описание	Ограничение
warehouse_id	SERIAL	Уникальный идентификатор.	PK, > 0.
name	VARCHAR(100)	Название склада.	Размер поля 100, NOT NULL.
address	VARCHAR(200)	Адрес склада.	Размер поля 200, NOT NULL.

Для «Зона хранения» (таблица 1.2) было описаны характеристики, а также добавлено поле для хранения типа зоны хранения в виде дискретного значения.

Таблица 1.2 – Описание «Зона хранения»

Имя поля	Тип данных	Описание	Ограничения
storage_id	SERIAL	Уникальный идентификатор.	PK, > 0.
warehouse_id	INT	Идентификатор склада.	FK, NOT NULL.
capacity	INT	Вместимость зоны.	>= 0, NOT NULL.
zone_type	zone_type	Тип зоны (дискретный).	NOT NULL.
zone_name	VARCHAR(100)	Название зоны.	Размер поля 100, NOT NULL.

Для «Товар» (таблица 1.3) были описаны характеристики и добавлено поле для хранения возможных типов товара в виде дискретного значения.

Таблица 1.3 – Описание «Товар»

Имя поля	Тип данных	Описание	Ограничения
product_id	SERIAL	Уникальный идентификатор.	PK, > 0.
name	VARCHAR(100)	Наименование товара.	Размер поля 100, NOT NULL.
expiry_date	DATE	Срок годности.	
product_type	product_type	Тип товара (дискретный).	NOT NULL.
is_active	BOOLEAN	Активен ли товар.	NOT NULL, DEFAULT TRUE.
photo	BYTEA	Фотография товара.	Никаких ограничений не накладывается.

Для «Поставщик» (таблица 1.4) были описаны характеристики.

Таблица 1.4 – Описание «Поставщик»

Имя поля	Тип данных	Описание	Ограничения
supplier_id	SERIAL	Уникальный идентификатор.	PK, > 0.
company_name	VARCHAR(100)	Название компании.	Размер поля 100, NOT NULL.
contact_person	VARCHAR(100)	Контактное лицо.	Размер поля 100.
phone	VARCHAR(20)	Телефон.	Размер поля 20.
address	VARCHAR(200)	Адрес компании.	Размер поля 200.

Для «Персонал» (таблица 1.5) были описаны характеристики.

Таблица 1.5 – Описание «Персонал»

Имя поля	Тип данных	Описание	Ограничения
employee_id	SERIAL	Уникальный идентификатор.	PK, > 0.
photo	BYTEA	Фотография сотрудника.	Никаких ограничений не накладывается.
full_name	VARCHAR(100)	ФИО сотрудника.	Размер поля 100, NOT NULL.
position	VARCHAR(50)	Должность.	Размер поля 50, NOT NULL.
phone	VARCHAR(20)	Номер телефона.	Размер поля 20.

Для «Учет товара» (таблица 1.6) были описаны характеристики.

Таблица 1.6 – Описание «Учет товара»

Имя поля	Тип данных	Описание	Ограничения
productAcc_id	SERIAL	Уникальный идентификатор.	PK, > 0.
supply_id	INT	Идентификатор поставки.	FK, NOT NULL.
employee_id	INT	Идентификатор сотрудника.	FK, NOT NULL.
storage_id	INT	Идентификатор зоны хранения.	FK, NOT NULL.
accounting_date	DATE	Дата учета.	NOT NULL.
quantity	INT	Количество товара.	>= 0, NOT NULL.
last_movement_date	DATE	Дата последнего перемещения.	Никаких ограничений не накладывается.

Также необходимо описать таблицу «Поставка» (таблица 1.8) для хранения связи М:М.

Таблица 1.8 – Описание «Поставка»

Имя поля	Тип данных	Описание	Ограничения
supply_id	SERIAL	Уникальный идентификатор.	PK, > 0.
product_id	INT	Идентификатор товара.	FK, NOT NULL.
supplier_id	INT	Идентификатор поставщика.	FK, NOT NULL.
supply_date	DATE	Дата поставки.	NOT NULL.
quantity	INT	Количество товара.	> 0, NOT NULL.

1.2 Создание БД

Для создания БД мы будем использовать систему управления БД (СУБД) PostgreSQL и pgAdmin. Она позволяет создавать таблицы как с помощью пользовательского кода, так и SQL кода и пользовательского интерфейса. Для нашего случая будем использовать SQL запросы и пользовательский интерфейс.

Начнём создание БД с её объявления (рисунок 1.2 и рисунок 1.3).

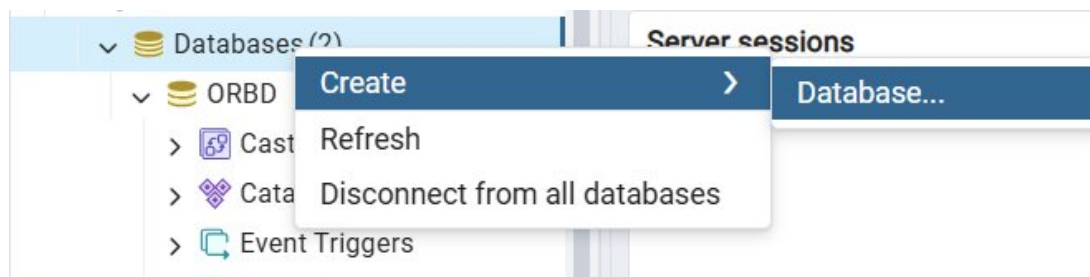


Рисунок 1.2 – Создание БД для склада

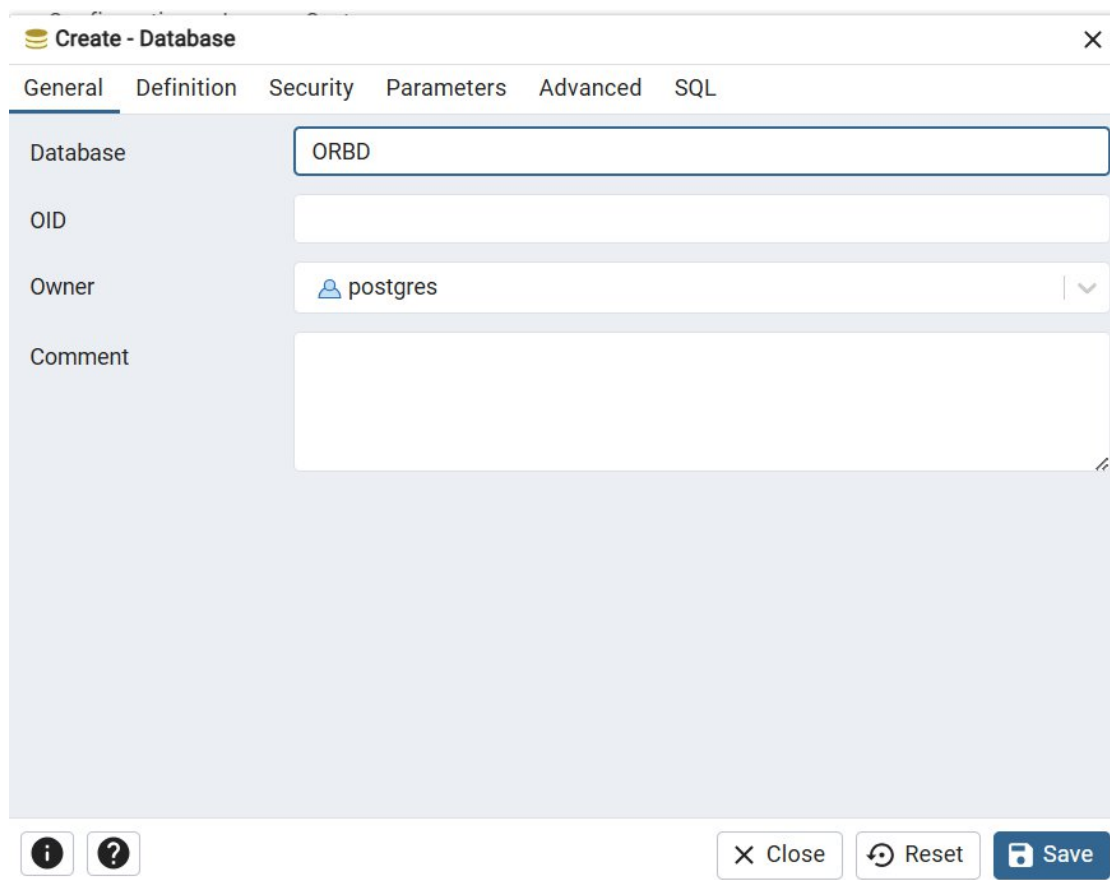


Рисунок 1.3 – Создание БД для склада

Как мы видим, наша БД отобразилась в интерфейсе pgAdmin (рисунок 1.3), и мы можем начинать работу с ней.

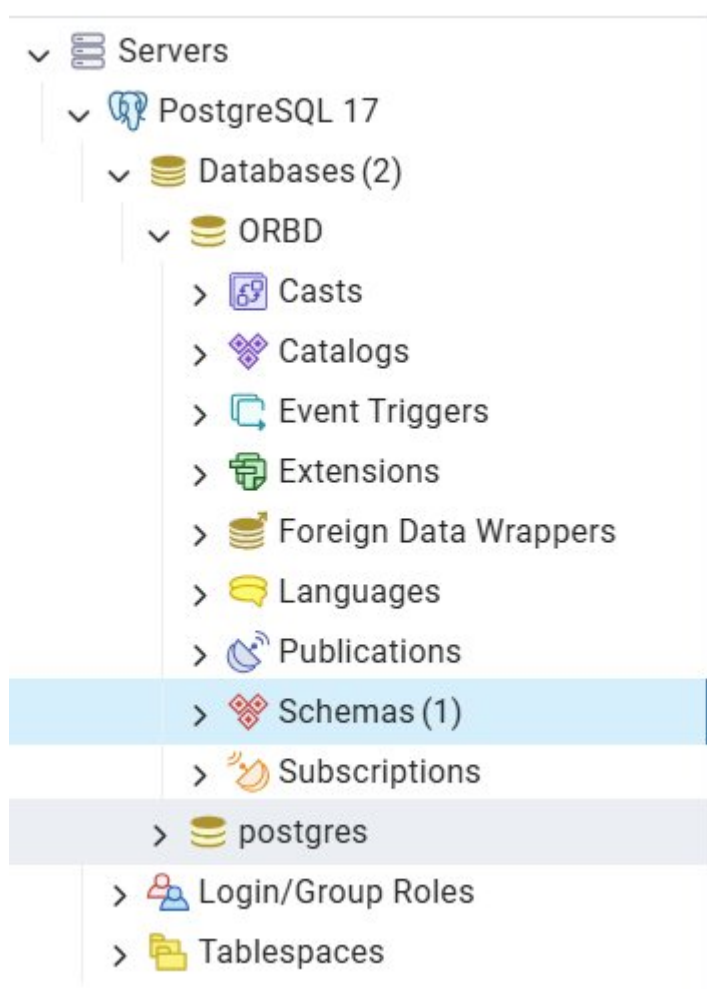


Рисунок 1.3 – Отображение БД в интерфейсе

Теперь напишем код для создания таблиц в нашей БД и связей между ними.

Листинг 1.1 – Создание таблиц

```
-- Создание типа ENUM для дискретных значений

CREATE TYPE product_type AS ENUM ('food', 'electronics',
'clothing', 'other');

CREATE TYPE zone_type AS ENUM ('refrigerated', 'dry', 'frozen',
'general');

-- Таблица "Склад"
CREATE TABLE warehouse (
    warehouse_id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    address VARCHAR(200) NOT NULL,
    CONSTRAINT warehouse_id_positive CHECK (warehouse_id > 0)
);

-- Таблица "Зона хранения"
CREATE TABLE storage_zone (
    storage_id SERIAL PRIMARY KEY,
    warehouse_id INT NOT NULL,
    capacity INT NOT NULL CHECK (capacity >= 0),
    zone_type zone_type NOT NULL,
    zone_name VARCHAR(100) NOT NULL,
    CONSTRAINT storage_id_positive CHECK (storage_id > 0),
    CONSTRAINT fk_warehouse FOREIGN KEY (warehouse_id)
REFERENCES warehouse(warehouse_id) ON DELETE RESTRICT
);

-- Таблица "Товар"
CREATE TABLE product (
    product_id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    expiry_date DATE,
    product_type product_type NOT NULL,
    is_active BOOLEAN NOT NULL DEFAULT TRUE,
```

```

    photo BYTEA,
    CONSTRAINT product_id_positive CHECK (product_id > 0)
);

-- Таблица "Поставщик"
CREATE TABLE supplier (
    supplier_id SERIAL PRIMARY KEY,
    company_name VARCHAR(100) NOT NULL,
    contact_person VARCHAR(100),
    phone VARCHAR(20),
    address VARCHAR(200),
    CONSTRAINT supplier_id_positive CHECK (supplier_id > 0)
);

-- Таблица "Персонал"
CREATE TABLE employee (
    employee_id SERIAL PRIMARY KEY,
    photo BYTEA,
    full_name VARCHAR(100) NOT NULL,
    position VARCHAR(50) NOT NULL,
    phone VARCHAR(20),
    CONSTRAINT employee_id_positive CHECK (employee_id > 0)
);

-- Таблица "Поставка"
CREATE TABLE supply (
    supply_id SERIAL PRIMARY KEY,
    product_id INT NOT NULL,
    supplier_id INT NOT NULL,
    supply_date DATE NOT NULL,
    quantity INT NOT NULL CHECK (quantity > 0),
    CONSTRAINT supply_id_positive CHECK (supply_id > 0),
    CONSTRAINT fk_product FOREIGN KEY (product_id) REFERENCES
product(product_id) ON DELETE RESTRICT,

```

```

        CONSTRAINT fk_supplier FOREIGN KEY (supplier_id) REFERENCES
supplier(supplier_id) ON DELETE RESTRICT
);

-- Таблица "Учет товара"
CREATE TABLE product_accounting (
    productAcc_id SERIAL PRIMARY KEY,
    supply_id INT NOT NULL,
    employee_id INT NOT NULL,
    storage_id INT NOT NULL,
    accounting_date DATE NOT NULL,
    quantity INT NOT NULL CHECK (quantity >= 0),
    last_movement_date DATE,
    CONSTRAINT productAcc_id_positive CHECK (productAcc_id > 0),
    CONSTRAINT fk_supply FOREIGN KEY (supply_id) REFERENCES
supply(supply_id) ON DELETE RESTRICT,
    CONSTRAINT fk_employee FOREIGN KEY (employee_id) REFERENCES
employee(employee_id) ON DELETE RESTRICT,
    CONSTRAINT fk_storage FOREIGN KEY (storage_id) REFERENCES
storage_zone(storage_id) ON DELETE RESTRICT
);

```

Теперь мы можем увидеть отображение всех таблиц в интерфейсе (рисунок 1.4) и на ERD схеме данных (рисунок 1.5).

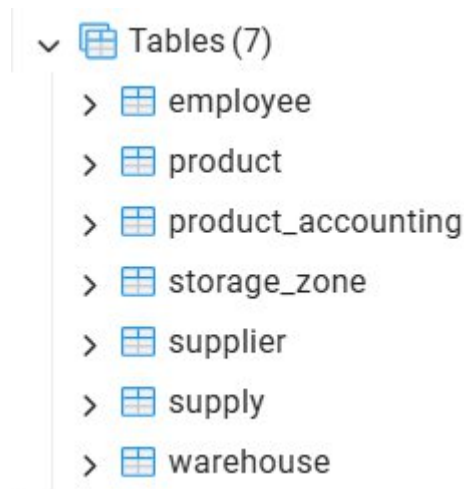


Рисунок 1.4 – Отображение таблиц в интерфейсе

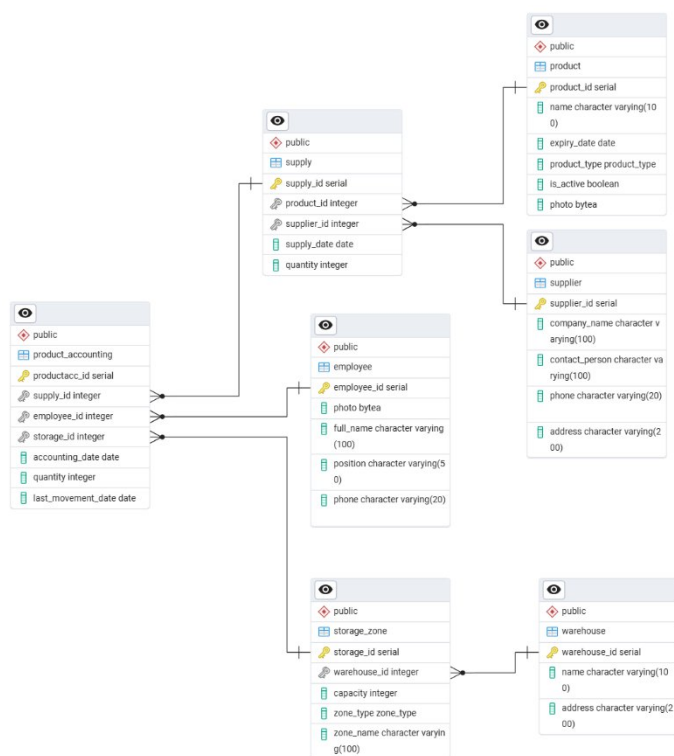
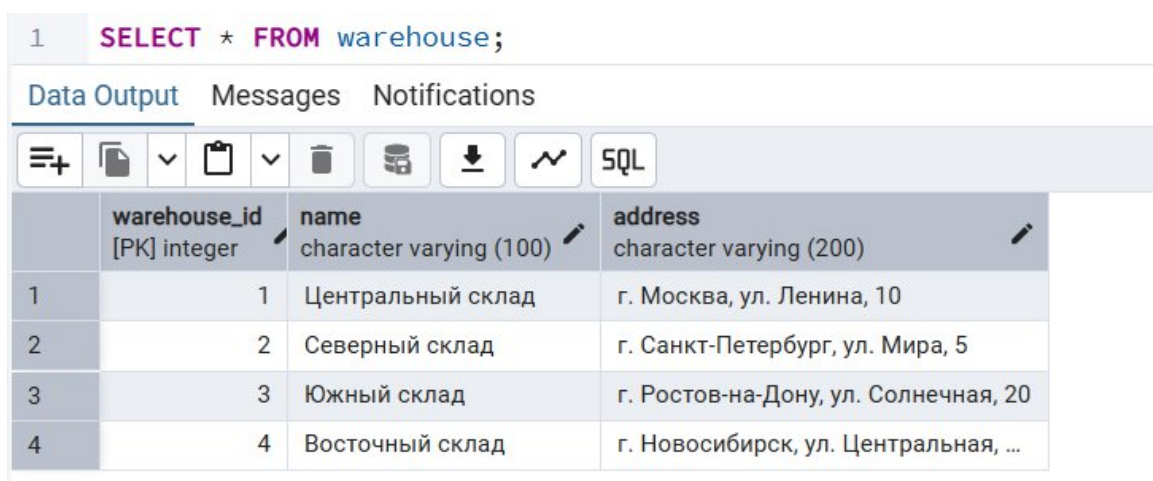


Рисунок 1.5 – ERD схема данных

Теперь мы можем писать запросы (рисунок 1.6-1.13) к БД, в которую заранее записали данные.

1 **SELECT** * **FROM** warehouse;

Data Output Messages Notifications

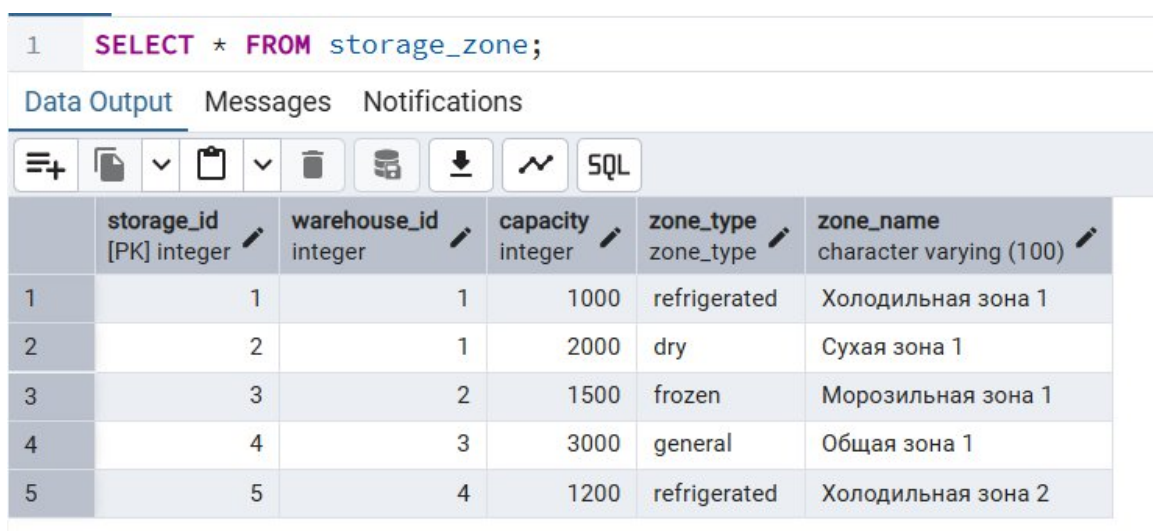


	warehouse_id [PK] integer	name character varying (100)	address character varying (200)
1	1	Центральный склад	г. Москва, ул. Ленина, 10
2	2	Северный склад	г. Санкт-Петербург, ул. Мира, 5
3	3	Южный склад	г. Ростов-на-Дону, ул. Солнечная, 20
4	4	Восточный склад	г. Новосибирск, ул. Центральная, ...

Рисунок 1.6 – Получение таблицы «Склад»

1 **SELECT** * **FROM** storage_zone;

Data Output Messages Notifications



	storage_id [PK] integer	warehouse_id integer	capacity integer	zone_type zone_type	zone_name character varying (100)
1	1	1	1000	refrigerated	Холодильная зона 1
2	2	1	2000	dry	Сухая зона 1
3	3	2	1500	frozen	Морозильная зона 1
4	4	3	3000	general	Общая зона 1
5	5	4	1200	refrigerated	Холодильная зона 2

Рисунок 1.7 – Получение таблицы «Зона хранения»

1 **SELECT** * **FROM** product;

Data Output Messages Notifications

SQL

	product_id [PK] integer	name character varying (100)	expiry_date date	product_type product_type	is_active boolean	photo bytea
1	1	Молоко	2025-05-01	food	true	[null]
2	2	Смартфон	[null]	electronics	true	[null]
3	3	Футболка	[null]	clothing	true	[null]
4	4	Книга	[null]	other	false	[null]
5	5	Яблоки	2025-06-01	food	true	[null]

Рисунок 1.8 – Получение таблицы «Товар»

1 **SELECT** * **FROM** supplier;

Data Output Messages Notifications

SQL Showing rows: 1 to 4 Page 1

	supplier_id [PK] integer	company_name character varying (100)	contact_person character varying (100)	phone character varying (20)	address character varying (200)
1	1	Молочная ферма	Иванов Иван	+79991234567	г. Москва, ул. Полянка, 12
2	2	ТехноТрейд	Петров Петр	+79992345678	г. Санкт-Петербург, ул. Невская,...
3	3	Модный мир	Сидорова Анна	+79993456789	г. Ростов-на-Дону, ул. Весенняя, 3
4	4	Книжный дом	Козлов Михаил	+79994567890	г. Новосибирск, ул. Книжная, 7

Рисунок 1.9 – Получение таблицы «Поставщик»

1 **SELECT** * **FROM** employee;

Data Output Messages Notifications

SQL Showing

	employee_id [PK] integer	photo bytea	full_name character varying (100)	position character varying (50)	phone character varying (20)
1	1	[null]	Смирнова Елена	Кладовщик	+79995678901
2	2	[null]	Кузнецов Алексей	Менеджер	+79996789012
3	3	[null]	Васильева Ольга	Кладовщик	+79997890123
4	4	[null]	Морозов Дмитрий	Логист	+79998901234
5	5	[null]	Лебедева Мария	Кладовщик	+79999012345

Рисунок 1.10 – Получение таблицы «Персонал»

1SELECT * FROM supply;

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	supply_id [PK] integer	product_id integer	supplier_id integer	supply_date date	quantity integer
1	1	1	1	2025-04-01	1000
2	2	2	2	2025-04-02	500
3	3	3	3	2025-04-03	200
4	4	4	4	2025-04-04	300
5	5	5	1	2025-04-05	1500

Рисунок 1.11 – Получение таблицы «Поставка»

1SELECT * FROM product_accounting;

Data Output

Messages

Notifications

</

Рисунок 1.12 – Получение таблицы «Учет товара»

Query Query History

1

SELECT * FROM public.storage_zone

2

ORDER BY storage_id ASC

Data Output Messages Notifications

Showing rows: 1 to 5

Pa

	storage_id [PK] integer	warehouse_id integer	capacity integer	zone_type zone_type	zone_name character varying (100)
1	1	1	1000	refrigerated	Холодильная зона 1
2	2	1	2000	dry	Сухая зона 1
3	3	2	1500	frozen	Морозильная зона 1
4	4	3	3000	general	Общая зона 1
5	5	4	1200	refrigerated	Холодильная зона 2

Рисунок 1.13 – Тестовый запрос к БД

БД проверяет на корректность ввод данных при добавлении элементов (рисунок 1.14), согласно выставленным ограничениям.

21	-- 1. Проверка NOT NULL для name
22	INSERT INTO warehouse (name, address) VALUES
23	(NULL, 'г. Тест, ул. Тестовая, 3');
24	-- Ожидаемая ошибка: column "name" of relation "warehouse" does not allow null values
Data Output Messages Notifications	
ERROR: значение NULL в столбце "name" отношения "warehouse" нарушает ограничение NOT NULL	
Ошибочная строка содержит (11, null, г. Тест, ул. Тестовая, 3).	
ОШИБКА: значение NULL в столбце "name" отношения "warehouse" нарушает ограничение NOT NULL	
SQL state: 23502	
Detail: Ошибочная строка содержит (11, null, г. Тест, ул. Тестовая, 3).	

Рисунок 1.14 – Проверка добавления элементов

Заключение

В ходе выполнения лабораторной работы была спроектирована и реализована реляционная база данных, отражающая структуру школьной информационной системы. Были созданы основные таблицы, такие как employee, product, product_accounting, storage_zone, supplier, supply и warehouse, а также определены связи между ними с использованием внешних ключей.

Особое внимание было уделено применению ограничений целостности (PRIMARY KEY, FOREIGN KEY, NOT NULL), что обеспечило корректность и надёжность хранения данных. Для проверки функционирования базы данных были добавлены тестовые и некорректные данные, что позволило убедиться в правильности реализованных ограничений.

Таким образом, цель лабораторной работы была достигнута: приобретены теоретические знания и практические умения по проектированию, созданию и тестированию реляционных баз данных.